T 00783225/9

00783225/9
DIALOG(R)File 349:PCT FULLTEXT

00783225 **Image available**
SYSTEM AND METHOD FOR SHARING COMPUTER ACTION SCRIPTS THROUGH A SEARCHABLE DATABASE, AND RELATED APPLICATIONS THEREOF
SYSTEME ET PROCEDE PERMETTANT LA MISE EN COMMUN DES SCRIPTS DES ACTIONS INFORMATISEES PAR L'INTERMEDIAIRE D'UNE BASE DE DONNEES CONSULTABLE, ET APPLICATIONS ASSOCIEES

Patent Applicant/Assignee:
VIRGINIA TECH INTELLECTUAL PROPERTIES INC, Suite 1625, 1872 Pratt Drive, Blacksburg, VA 24060, US, US (Residence), US (Nationality)
Inventor(s):
JONES Mark T, 900 Highland Circle, Blacksburg, VA 24060, US,
COUPEY Eloise, 900 Highland Circle, Blacksburg, VA 24060, US,
Legal Representative:
FLOAM D Andrew (et al) (agent), Needle & Rosenberg, P.C., Suite 1200, The Candler Building, 127 Peachtree Street N.E., Atlanta, GA 30303-1811, US

Patent and Priority Information (Country, Number, Date):
Patent: WO 200116765 A1 20010308 (WO 0116765)
Application: WO 2000US23749 20000830 (PCT/WO US0023749)
Priority Application: US 99151927 19990901
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CR CU CZ DE DK DM DZ EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT TZ UA UG UZ VN YU ZA ZW
(EP) AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE
(OA) BF BJ CF CG CI CM GA GN GW ML MR NE SN TD TG
(AP) GH GM KE LS MW MZ SD SL SZ TZ UG ZW
(EA) AM AZ BY KG KZ MD RU TJ TM
Main International Patent Class (v7): G06F-013/00
Publication Language: English
Filing Language: English
Fulltext Word Count: 12657

English Abstract
Software, system and method for sharing scripts of computer actions. Scripts representing actions that can be taken by a device (200, 210) are stored in a searchable database (110). The scripts represent device actions, and particularly, actions that involve using or accessing information from the Internet (300). Users can search and retrieve scripts in the scripts database (100) from a device (200, 210) to perform computer actions on the devices (200, 210). A database server (100, 110) stores and manages access to the scripts database (110). Search queries are processed by middleware, called ScriptEngine middleware (410), and submitted to the database server (100, 110). The ScriptEngine middleware (410) returns scripts to applications resident on a user device (200, 210), and the applications interpret the scripts for performing device operations.

French Abstract
L'invention concerne un logiciel, un systeme et un procede permettant la mise en commun des scripts d'actions informatisees. Les scripts

representant les actions pouvant etre executees par un dispositif (200, 210) sont memorises dans une base de donnees (110) consultable. Ces scripts representent les actions executees par un dispositif, et en particulier les actions consistant a utiliser ou a solliciter une information dans Internet (300). Les utilisateurs peuvent rechercher et recuperer des scripts dans la base de donnees (100) a partir d'un dispositif (100, 210), afin d'executer des actions informatisees sur le dispositif (200, 210). Un serveur (100, 110) de base de donnees memorise et gere l'acces a la base de donnees (110) de scripts. Les demandes de recherche sont traitees par un logiciel (410) intermediaire appele ScriptEngine, et soumises au serveur (100, 110) de la base de donnees. Le logiciel (410) intermediaire ScriptEngine retourne des scripts aux applications residantes d'un dispositif (200, 210) utilisateur, et ces applications interpretent ces scripts afin d'executer des operations informatisees.

Legal Status (Type, Date, Text)

Publication	20010308	A1 With international search report.
Examination	20010809	Request for preliminary examination prior to end of 19th month from priority date
Correction	20020906	Corrected version of Pamphlet: pages 1/14-14/14, drawings, replaced by new pages 1/14-14/14; due to late transmittal by the receiving Office
Republication	20020906	A1 With international search report.

Detailed Description

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes andAbbreviations " appearing at the beginning of each regular issue of the PCT Gazette.

SYSTEM AND METHOD FOR SHARING COMPUTER ACTION SCRIPTS THROUGH A SEARCHABLE DATABASE, AND RELATED APPLICATIONS THEREOF

This application claims priority to U.S. Provisional Application No. 60/151,927, filed September 1, 1999, the entirety of which is incorporated herein by reference.

FIELD OF THE INVENTION

The present invention relates to sharing scripts related to computer device actions useful, for example, in automating user tasks on the WWW and the Internet.

BACKGROUND OF THE INVENTION

Use of the World Wide Web (WWW) and the Internet has been characterized 1 5 by "surfing" of the available resources. Graphical User Interfaces (GUI's), such as browsers, have been the primary user interface to the WWW and the Internet. Such interfaces require that users initiate every action that they want to take. For example, visiting a web site requires clicking on the interface, and searching the WWW requires clicking to visit a particular web page and then entering the desired keywords. While such an interface is appropriate for "surfing the Web," it does not provide the means to efficiently execute tasks, particularly tasks that are repeated on a regular basis by one or more users. Efficient use of the WWW and the Internet requires a method for automating tasks without requiring unnecessary user intervention or knowledge. Such an interface is useful, for example, on a personal digital assistant (PDA), as well as for the visually impaired using desktop computers.

Automating user tasks on the Internet reduces the burden on users to recall the details of every operation, thus creating greater ease-of-use

and increased efficiency by reducing the amount of time wasted forgetting how to execute tasks. For example, the use of bookmarks on WWW browsers such as Netscape Navigator (America Online)
"D
1
and Internet Explorer (Microsoft) allows users to keep a list of favorite WWW sites and to access them with a single click without having to remember the LTRL address for every favorite site. Unfortunately, this relatively simplistic type of automation requires that users have access to the files containing these bookmarks, which typically reside on a single computer and may not be accessible, for example, from a PDA.

Automation can also save users effort, rather than just reducing the burden on memory. For example, users shopping on the Internet typically have to type in their shipping address, billing address, credit card information, and other details when making a purchase. This excessive typing is often such a barrier to purchase that users leave the site, rather than finalize their purchases. Fortunately, such information rarely changes for an individual user. At least two types of automated approaches have been developed to address this situation. In one approach, Microsoft's Internet Explorer recognizes that certain WWW pages are prompting a user for this information and will enter the information automatically for the user (assuming that the information has been given to Internet Explorer at an earlier time). Another keystroke-saving approach is the 5 "One-Click" service at Amazon.com. See U.S. Patent No. 5,960,41 1. The one-click functionality allows users to enter the information one time at a web site and then use a single click to confirin that infonnation for later visits to the site. Variants of this approach abound on the WWW. A shortcoming of these approaches, however, is that they only apply to filling out forms on the WWW, and not to more general tasks on the WWW and the Internet.

Software for automating user tasks on a single computer exists. Such software typically takes the form of keystroke macros. A user uses the software to "record" a sequence of keystrokes as a macro. This macro can be executed by a single keystroke, or by a small number of keystrokes at a later time, saving the user the effort of entering the keystrokes, as well as the difficulty of remembering them. Although such an approach can be very useful on a single computer, it has the significant drawback that it WO 01/16765 PCT/USOO/23749
3
files/directories named differently), the keystroke macro will not work. In addition, like bookmarks, these macros exist in a single file and are not typically accessible on a variety of computers. Such an approach cannot provide a seamless environment for a mobile user with a personal computer, a PDA, a cellular phone, or other Internet appliances.

Higher-level approaches to automating tasks have been developed and used in computing environments by expert users for many years. These approaches typically take the form of scripting languages that allow a user to describe a complex set of tasks and then execute them as a single command. Examples of scripting languages include JavaScript (Sun Microsystems), VBScn'pt (Microsoft), and UNIX shell scripts (e.g., numerous authors, including Bell Labs). Such scripting languages operate at a slightly higher level than programming languages such as the "C" language, the FORTRAN language, or the JAVA language. These languages often are accompanied by a method for recording a script. For example, many UNIX shells provide a command that allows users to record their commands into a file until further notice. Once these commands are recorded to a file, they can replayed at a later date by "executing" that

file, thereby saving the time to retype and remember all of the commands. If carefully written in such a form as to avoid fixing particular filenames and/or locations of programs, such scripts can achieve a measure of portability across computers that support that scripting language - certainly achieving more portability than typical programming language and/or keystroke macros. Such scripting languages are powerful tools in the hands of expert users, but are too complex for most users. They are computer languages that require computer programming skills to understand and operate them. Further, the operation that a particular script will perform is not obvious, and it is often difficult for a novice to find the right script for his/her task (if such a script exists). In particular, there is no mechanism for a novice to search for the correct script, or to have the correct script suggested to him/her.

The nearly universal connectivity provided by the Internet allows data to be available to users from many locations without having physical access to the computer that stores that information. Two examples of this connectivity are the Instant Messenger program (America Online) and customized WWW portals, such as that provided by Yahoo! The Instant Messenger program gives users the ability to monitor which members of their "buddy list" are online, and to send short messages to users on this list. The Instant Messenger service is available on a variety of platforms. The user will see the same "buddy list" no matter where he or she uses the service because the information is not stored on a local computer, but rather on servers at America Online.

The customized portal available at "My Yahoo!" allows a user to specify which inforination he/she would like displayed on a personal portal WWW page. This information is selected from a list of possibilities provided by Yahoo! (e.g., weather for a specific locale, news, and the recent results from a user's favorite sports teams). This portal is available from any WWW browser connected to the Internet because the 1 5 user's configuration is stored at Yahoo! rather than on a local computer. By capitalizing on the connectivity of the Internet, both of these services provide portability, but neither provides the task automation service required for efficient use of the Internet.

The proliferation of devices that can access the Internet and the WWW, including PDA's, cellular phones, and Internet appliances, provides users with the ability to access data from many locations in a very flexible fashion. Unfortunately, the physical user interface on these devices differs greatly from that of a desktop personal computer. For example, a PDA has a small screen and a pen-based interface, making it
:n
unsuitable for displaying most Web pages and very difficult to use to fill out forms.

The Universal Interface Markup Language (UIML) is a language that allows a user interface to be described in a fashion that is independent of the physical user interface.

A web page written in UIML (as opposed to HTML) would be viewed in a fashion appropriate to the particular device currently being used by the user. An alternative approach currently in commercial use is the development of a Web browser appropriate for PDA's, combined with "clipping" services that reduce the content on particular Web pages to a size appropriate for PDA's (e.g., the browser provided by AvantGo).

While both of these approaches are steps towards a universal interface, they are not steps towards automating tasks. Instead, they focus on

providing the ability to "surf the Web" with several different devices.

Notwithstanding the above-described advantages and strong need, no significant user task automation system, method, or software has been widely adopted for the WWW and the Internet, primarily because no existing method provides all of the qualities required for users to embrace the technology. Such a scripting system would have to automate high-level tasks (e.g., searching the web), allow transparent use from a variety of platfonns including PCs, PDA's, cellular phones, and other Internet appliances, express tasks in a form that any user can understand and manipulate, allow
ily exchange task descriptions (scripts), provide a mechanism to advise users to easi 1
1 5 users on appropriate scripts to execute, and allow users to easily find the right script for their tasks.

SUMMARY OF THE INVENTION
The present invention is directed to software, a system and a method for sharing scripts of computer actions. Scripts representing actions that can taken by a device are stored in a searchable database. The scripts represent actions: particularly actions that involve using or accessing information from the Internet. Users search and retrieve scripts in the scripts database using a device (e.g., PC, PDA) to perform device operations based on script actions. A database server stores and manages access to the scripts database. Search queries are processed by middleware, called ScriptEngine middleware, and submitted to the database server. The ScniptEngine middleware returns scripts to applications resident on a user device. The applications of the devices interpret the script for performing device operations.
The system and method of the present invention greatly automates user tasks and is well suited for tasks involving the Intemet/WWW. The scripts represent higherlevel device functionality, and therefore allow transparent use from a variety of platforms, including PC's, PDA's, cellular phones, and other Internet appliances. For example, the scripts automate tasks such as accessing a web page, searching the web through a portal, etc. The scripts express tasks in a forrn that any user can understand and manipulate, and they allow users to readily share tasks, easily find the night script for their tasks, and readily advise users on appropriate scripts to execute. The ease and flexibility of script creation makes the scripting function accessible to a wide range of users who vary greatly in computer and programming skills. In addition, the front-end applications that reside on user devices can be designed to interpret scripts in a manner compatible with device characteristics and/or user characteristics.

The above and other objects and advantages of the present invention will become more readily apparent when reference is made to the following description 1 5 taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS
FIG. 1 is a block diagram of a hardware architecture of the system according to the present invention.

FIG. 2 is a diagram showing a software architecture of the system according to the present invention.

FIG. 3 is a diagram showing a scalable architecture of the system according to the present invention.

FIG. 4 is a screen shot showing an example of a main graphical user

interface (GUI) provided according to one embodiment of the present invention.

FIG. 5 is a table that represents one possible organization of a database of scripts according to the present invention.

FIG. 6 is a screen shot showing an example of a script search GUI that allows a user to search for scripts in the database using a variety of criteria.

FIG. 7 is a screen shot showing an example of a search results GUI that allows a user to view the scripts returned by a search of the database and select them for execution.
FIG. 8 is a screen shot showing an example of a script edit/execute GUI that allows a user edit the script to modify actions as well as execute the script.

FIG. 9 is a screen shot showing an example of a remote script GUI that allows a user to send a script to another user for execution.

FIG. 1 0 is a screen shot showing an example of an action advisor GUI that suggests a script to a user, providing the user with the capability to execute the suggested scripts and/or modify the criteria the action advisor uses to suggest scripts.

FIG. I I illustrates a table that represents an example of an organization of a database of script access records.

1 5 FIG. 12 illustrates a table that represents an example of an organization of a database of users.

FIG. 13 illustrates a GUI that allows a user to send scripts to other users.

FIG. 14 illustrates a table that represents exemplary script actions for purposes of explaining how these actions may be interpreted differently by front-end applications on different types of devices.

FIGs. 15 and 16 are diagrams depicting how front-end applications interpret script actions differently to perforin different device operations.

DETAILED DESCRIPTION OF THE INVENTION
The present invention now will be described more fully hereinafter with reference to the accompanying drawinars, in which one embodiment of the invention is shown. This invention may, however, be embodied in many different forms and should not be construed as limited to the embodiments set forth herein; rather, applicant provides these embodiments so that this disclosure will be thorough and complete, and iII fully convey the scope of the invention to those skilled in the art. Like numbers
wi I
refer to like elements throughout.

Referring first to FIG. 1, one configuration of the hardware components of a system according to the present invention is shown at reference numeral 10. The system IO comprises a database server computer(s) 1 00, a plurality of computer devices 200 (hereinafter also simply referred to as "devices"), a communication network(s) 300 and a ScriptEngine middleware computer(s) 400. The communication network(s) 300 may comprise the Internet, local area networks, wireless networks, etc., or any

combinations thereof, as is known in the art. For example, the database server computer(s) I 00 and/or the ScriptEngine middleware computer(s) 400 may be part of the hardware that operates as a WWW site that is accessed by users of the devices 200 through the WWW.

The database server computer(s) I 00 stores the scripts in a database. The 1 5 ScriptEngine middleware computer(s) 400 receives access requests for scripts from devices 200 (via the communication network(s) 300) and forwards them to the database server computer(s) I 00. Similarly, the ScriptEngine middleware computer(s) 400 receives the scripts from the database server computer(s) 400 and forwards them to the devices 200. In each device 200, there is at least one front-end application that acts on a script. As used hereinafter, the term "script" means instructions that represent the execution of one or more computer device actions. An "action" is defined as a single, discrete, user-initiated activity on a device 200.

The devices 200 may comprise any computer device known or hereinafter developed, including, for example, desktop computers, laptop computers, hand-held computers (PDA's such as PaIMTm devices), hand-held computers with wireless communication capability to the Internet, web browsers, Internet appliances, and other wireless devices, mobile telephones, etc. As will be explained further hereinafter, there are particular utilities of the present invention in the case of hand-held wireless devices that typically have different user interfaces from desktop and laptop computers.

There are alternatives to the hardware architecture shown in FIG. 1. For example, the database server computer(s) 100 and the ScriptEngine middleware computer(s) may be embodied on one computer (or a common set of computers).

Furthermore, the functions of the device front-end applications, database server and ScriptEngine middleware may be executed on a single computer device, providing a fully dedicated (and isolated) configuration.

Refeming now to FIG. 2, a software architecture of the system is shown. This is a three-tier architecture, though other architectures are possible. The first tier is the user (front-end) applications 2 1 0. The second tier of the architecture is the ScriptEngine middleware 41 0, and the third tier is the database server software I IO.
The ScriptEngine middleware 410 is responsible for accepting requests from the applications 21 0 and translating these requests into queries for the database server software I IO. The ScriptEngine 41 0 is also responsible for processing the records returned from the database server and returning these records as scripts to the application 2 1 0. The database server software 1 1 0 controls access to a script database (an example of which is depicted in FIG. 5 described hereinafter), the script access database (an example of which is depicted in FIG. I I described hereinafter), and the user information database (described later and depicted in FIG. 12 described hereinafter).

FIG. 2 also shows several types of applications 21 0, including a business process application 212, a MyScn'pt user application 214, a PDA/wireless application 216, a sight-impaired content translation application 218, and a team decision application 220. These are only examples of possible applications, and it should be understood that there are an unlimited number of other applications that are useful in accordance with the system and method of the present invention.

The three-tier architecture allows for many different front-end (and types of front-end) applications 210 that access the ScriptEngine middleware 410. The ScriptEngine middleware 41 0 does not interpret the scripts; rather it accepts requests and sends them on to the database server software I 10. This organization allows multiple databases to exist behind the middleware, with the access controlled by the middleware. FIG. 3 illustrates a variation of the architecture that is fully scalable to many users because many instances of the ScriiptEngine middleware 410 can be running to support many applications 21 0. Similarly, the database server software 1 1 0 can be scalable commercial database server software that can support many 1 0 transactions per second. In addition, the database servers I IO can be partitioned among users and different types of applications to further increase scalability. That is, all ScriptEngine middleware (servers) 410 do not have to communicate with all database servers I I 0. Consequently, there are many more applications 21 0 than there database servers 1 1 0.

1 5 Alternatives to this architecture include compressed versions of the three-tier architecture. For example, applications 21 0 could be folded into the same program as the ScriptEngine middleware 41 0, thus creating a client/server architecture. Another alternative is a system in which the database server I IO and ScriptEngine middleware 410 are executed on a user device. This would decrease network performance and improve privacy at the cost of tying the user's files (and scripts) to a single computer.
Another and more attractive alternative is a peer-to-peer script sharing architecture. In this architecture, users run the ScriptEngine middleware 41 0 and database server I IO on their own devices, but advertise their servers (via an exchange server run by a third party) to their friends/family members/associates who are currently online. A user logs into the system by sending a message to the exchange server with his current IP address and port number along with a list of users with whom he wishes to communicate. The exchange server responds with the list of the requested users who are actually online along with the 1P address and port number of each of those users. When a user logs out of the system, a message is sent to the exchange server and forwarded to every user interested in that user. The exchange server is only responsible for maintaining the user information database such as depicted in FIG. 12 and described hereinafter; all script information is kept in the database local to a user's device. Such an organization is inherently scalable because the database inforillation is spread over all user devices. It does suffer from the drawback that users cannot search the database of other users without permission, and there is no global database of scripts. In addition, a user's files 'II tied to a single device. Still another alternative is for the Scn'ptEngine
are sti
'ddleware 410 and the database server software I 10 to run on the same computer or
ml
set of computer(s), as opposed to separate computers as shown in FIG. 1.

Operation of the system in the context of the team decision application 220 will now be described. Users have the team decision application 220 resident in their device. The team decision application 220 is a software application program that allows people to efficiently coordinate information from many different sources to make decisions. Decisions can be made using information from the Internet, files previously stored on a device, or even through real-time communications with other members of a decision team.

The team decision application provides users with a graphical user interface (GUI) such as the one shown depicted in FIG. 4. The GUI provides the users with a set of actions they can execute. A list of user actions is shown in Table 1 below.

Table I
Communication Tools Script Option
Whiteboard Select Communication Edit/SaveRecorded Data Record Actions Group
Group Discussion Open Browser Load/Edit/Execute Scripts Provide Suggestions
Send Notice Search the Web Start Script for Other Users
Video Conference Play Audio
Exit Play Video
Examples of such actions include visiting a particular WWW site or searching the WWW using particular keywords. A unique aspect of the team decision application is its flexibility and usefulness when executing tasks associated with the WVV`W and the Internet. The program allows sequences of actions to be collected into a single script for future execution. These scripts, composed of actions, are intended to accomplish a user task. The scripts are stored in a searchable database (the script database) to allow flexible access to the scripts (FIGs. 1 and 2), as will be described further hereinafter.

Whether working together or independently to manage decision makinar, team members
O
can organize the actions they each take in making a decision in the form of scripts.

These scripts can be kept on file in the script database for future, repeated use. Any team member can send a created and stored script to other team members.

Each action in a script is a record in a database as depicted in FIG. 5. Fields in the database may include, but are not limited to, the name of the script, a description of the script, the name of the user who stored the script, the type of action, and the date the action was performed. Additional fields may include those specific to a particular option, for example, the URL to be visited if the action is visiting a Web page or the keywords if the action is searching the WWW. Multiple scripts are expected to be stored in the database, and multiple databases may be used to increase capacity or improve efficiency by segmenting the database according to users or applications, as explained above. Segmenting is possible with most commercial database software.

Users of the team decision application record their actions, which are then saved as scripts. To do so, a user turns on a "Record Action" button, perforins her actions, and turns off the "Record Action" button through a GUI such as that shown in FIG. 1.

All actions executed by the user during that time will be recorded. The user can then review these actions and save them with a given script name and description. The 'II be stored by the database server as records in a fon-nat similar to that shown
actions wi
in FIG. 5 with the given script name, username, date, and description.

Users of the team decision application can search the script database for an appropriate script to perform their task. A GUI suitable for a user to

perforrn such a search is shown in FIG. 6. The GUI allows the users to search via a logical combination of user name, date, action name, and script name. Any logical combination of fields in the database is possible, only requiring the appropriate user interface to express the search criteria. The GUI sends the search criteria to the ScriptEngine middleware (FIG. 3), which translates the search criteria into an SQL query to be sent to the database server, which in turn accesses the script database. The scripts found by the search are returned by the ScriptEngine middleware to the team decision application for presentation to the user via a GUI such as the one shown in FIG. 7. The user may then select the script for execution.

Users of the team decision application can edit and execute scripts. A script can be edited to eliminate actions, add actions, or modify actions in the script. For example, a keyword can be changed in a search-the-web action of a script. The editing of scripts can be accomplished in the team decision application via the GUI depicted in FIG. S. Scripts can be executed in part, step-by-step, or all at once. The execution of a script involves executing each of the actions of the script. For example, a new window with the specified Web page is brought up when an "open browser" action is executed or a message is sent to the given user when a "send notice" action is executed. These actions are executed by the local team decision application on a user device. It may be desirable to introduce user-specified time delays in between actions to execute actions at a particular pace (e.g., a slide show of Web sites). Such time delays would be actions in the script database where such an action simply waits for n seconds when the script is executed.

Users of the team decision application can send scripts to other users for remote execution using a GUI such as the one depicted in FIG. 9. This facility allows a user to share a script with friends; for example, sending a script that takes users to sites describing a recent vacation. Performing this task merely requires that one team decision application send the name of the script to another team member. The user receiving the script is notified and given the name of the script and the user sending the script. The receiving user may review the script prior to execute using the edit script capability. Once accepted for execution, this script executes like any other script.

Another functionality that is useful is an "Action Advisor" functionality. The Action Advisor (if enabled by a user) monitors the actions of a user, keeping track of the most recent set of actions. Based on those actions, it searches the script database for scripts that are similar to the user's recent actions, and then presents the most likely scripts to user for possible execution. The Action Advisor can reside in the ScriptEngine middleware or in the team decision application itself. When turned on, the Action Advisor works without further action on the user's part until scripts are presented to the user for possible execution. An algorithm for the Action Advisor is.

(1) Maintain a short list of the user's most recent actions.

(2) When the user executes an action, use the values in the fields in the most recent actions to search the script database for scripts whose fields match
those of the user's most recent actions. For example, each action has a name. The search can be for those scripts with actions that match the user's
most recent actions. The search can also be for those scripts whose username is the same. In essence, the search can be on any combination of

the fields in an action because the actions in the script database are stored under several. To perform this search, the Action Advisor sends the search request to the ScriptEngine middleware (or directly to the database server).

(3) Return the list of matching scripts to the Action Advisor which ranks the matching scripts for presentation to the user. Several criteria can be used to
determine the ranking of the scripts. The algorithm allows for different criteria to be used - each matching script is scored on each of the criteria, then a weighted combination of the score on each criterion is used to give a single figure of merit to each script. The weight assigned to each criterion
can be selected by the user. One possible figure of merit is a linear combination of the criteria. For example, the following equation has been implemented in the team decision application.

Merit Weight-I * (I - if the username is a match, 0 - otherwise) + Weight-2 * (I - if the actions are an exact match, 0 - otherwise) + Weight-3 * (the number of times the user has accessed this script)

(4) After the scripts have been scored and ranked, the Action Advisor presents to the user those scripts whose figure of merit score is greater than a cutoff value (the scripts are sorted by rank). Those scripts whose scores are below
the cutoff value are not presented to the user. The user may adjust this cutoff value. If no scripts meet the cutoff value, then none are presented to the user.

A depiction of the Action Advisor suggesting a script and then presenting a ranking of the matching scripts is shown in FIG. IO. The user can manipulate the weights for the figure of merit computation as well as set the cutoff for reporting scripts. This weiQhtiniz scheme is just an example of one of many possible user-adjustable schemes available.

An additional database (accessed via the ScriptEngine middleware and database server) is necessary to support the Action Advisor. This database is called a script access database and it keeps track of which scripts a user has accessed and how many times a user has accessed a particular script. FIG. 1 1 shows an example of a script access database containing this information. The script access database can track when a user last used the script, the number of times they have used the script, and the number of times they have rejected the script for execution when the Action Advisor or the search engine have suggested the script to the user. This information can be used in ranking a script for presentation to a user. In addition, the script access database can track the number of times all users access or reject a script.

A further utility of the Action Adviser is to monitor script actions executed by a user and/or scripts used by a user at a predeten-nined time of day and/or frequency, such as the first set of actions a user perfon-ns when turning on his/her computer. The Action Adviser detects this pattern and stores a script representing those actions in the scripts database, or if it is an existing script that is repeatedly executed, makes a note in the scripts access database about that script. The Action Advisor advises the user that the same actions (or script) are performed on a consistent basis and can be made to automatically occur on the user's machine at that time of the day and/or frequency. An automatic timer is set in the application associated with the script so that the script is automatically activated at a predetermined time and/or

frequency.

The team decision application presents a consistent view of information and functionality to a user no matter where the user is executing it. This consistent view is achieved because no permanent files are stored on the user's local device. Rather, all user information and script information are stored in a database (or databases).

As described in the preceding paragraphs, scripts are created by a user who records a sequence of actions in the team decision application and then requesting that those actions be saved to a script. When requested to save a script, (1) the team decision application prompts the user for a name and, in some cases, a description to save the script under; (2) the team decision application sends a message containing those actions to the ScriiptEngine middleware (unless the actions have already been sent to the ScriptEngine middleware); (3) the ScriptEngine middleware sends the actions to the database server to be saved; and (4) the database server saves the actions as records in the script database.

When a particular script is executed by the user, the script access database is updated to reflect the access date, and the access count is also updated. When a script is rejected for execution by the user (after being suggested by the Action Advisor or returned by a script search), the script access database is also updated. This update takes place by (1) the application sending a message to the ScriptEngine middleware with the access information; (2) the ScniptEngine middleware receiving this messages and translating it into SQL for the database server; and (3) the database server receiving the message from the ScriptEngine middleware and updating the script access database.

It may be desirable for the team decision application to maintain some level of secure access to scripts and confirmation of user identities. This security should allow users to protect the scripts they do not wish others to see. In addition, the system should be able to validate a user's identity, and ensure that other users can validate another user's identity. For example, if a user called "John Doe" sends a message requesting execution of script "XYZ," a user wants to be assured that it is indeed "John Doe" sending that message. To accomplish either task, the identity of a user must be firmly established. This can be accomplished by requiring a user to enter a unique identifier, for example a password. This unique identifier is kept in a user information database shown in FIG. 12 that is accessed by the database server. A stronger form of security can be provided by a public key encryption system (using one of several wellknown public key encryption methodologies). When a user logs into the system, he or she is required to provide a private key to the front-end application only. This private key is stored by the front-end application for the duration of the session (but never sent out by the front-end application). When any message, X, is sent to the ScriptEngine middleware requesting any service, it is encrypted in the following form. (1) The message X is first encrypted with the private key of the user to form P I (X).

(2) This encrypted message, PI (X), is then encrypted with the public key of the
ScriptEngine middleware to form P2(Pl(X)). Note that this message can only be decrypted by a user holding the private key of the ScriiptEngine middleware. This private key is not available to any user and is protected by the middleware.

(3) The encrypted message, P2(Pl(X)), is sent to the ScriptEngine

middleware.

is
(4) The ScniptEngine middleware decrypts the message using its private key to retrieve PI (A).

(5) The ScriiptEngine middleware queries the database server to retrieve the public key of the user who has this message"says" it has been sent from. If the message decrypts properly using that public key to forin X, then that
message is indeed from user who holds that private key (assuming no one has stolen that private key).

A similar technique can be used by the ScriptEngine middleware to verify to user A that user B is indeed who he represents himself to be.

With the ability to verify users' identities, it is a simple matter to protect scripts from other users. The script database can be augmented by a field that indicates whether a script is public or private. If a script is private, then it will only be returned to the user who saved the script (which is indicated in the username field of the script database). This access control can be expanded to access lists of users who may access 5 a particular script.

The present invention involves several other novel applications referred to above that go beyond those embodied in the team decision application.

A Consumer-oriented Application: MyScrip
The MyScript application 214 (FIG. 2) is an application designed to make the Internet more usable for individual users. At a basic level, the MyScript application connects users to the Internet, in much the same way that Instant Messenger and ICQ operate. Unlike those applications, however, MyScript also helps people complete Intemet-related tasks by enabling them to execute scripts in a fashion similar to the team decision application. These scripts can be drawn from a central database of scripts, or swapped with other users.

A wide array of actions can be automated with the MyScnipt application, including, but not limited to, the following actions.

In
Send a message/chat with other MyScnipt users
* See a site (a forin of remote scripting that enables one user to create a site tour
for another user)
+Look for a song (search music web sites for digital music files)
Create a playlist (create a set of digital music files based on user-specified
criteria)
*Get a movie (download a digital movie onto the user's computer)
* Get hardware/software tools (downloads tool, such as a printer driver or a
software patch, to the user's computer)
I 0 +Compare sites (enable a user to simultaneously view product information, in
the format created by the vendor, for up to four sites)
+Create personal profile (compile selected user information for online forms,
e.g., user registration, online shopping
+Update auction status (track expiring auctions and new auctions targeted

by
1 5 the user, update auctions in which user is a participant)
By combining these actions into scripts, the MyScript application makes it possible to automate the execution of complex tasks on the Internet. With the MyScript application, people build scripts, store scripts, retrieve scripts, and send scripts - all with an organized, easy-to-use, interface. These capabilities are described below.

Swapping Scripts with Other Users. To illustrate the concept of scriptswapping, suppose a user has compiled a set of songs in a particularly appealing sequence. With the MyScnipt application, a user can send a sequence of song files as a script to another MyScnipt application user. When executed, the MyScript application plays the list of songs for the other user. When another user receives these scripts via MyScript, he or she decides whether to execute the script. This swapping capability can be used among friends in much the same way that Instant Messenger (though AOL) is used. For example, a user may choose a group of other users from a list of available users (as shown in FIG. 13), and request that a particular script be sent to those users.

Creating Dynamic Content. A distinguishing characteristic of the MyScnipt application is that it creates dynamic content. Dynamic content is content that changes over time and is presented in a particular sequence. The execution of a script represents dynamic content because it causes information to be presented in a particular sequence, and that infori-nation is updated every time the script is executed.

For example, suppose a user's hobby is selling and collecting baseball cards.

I 0 The user searches for particular groups of cards that he/she would like to add to his/her collection. At present, cards on the Internet are sold on several different auction sites, as well as through individual vendors. The user would like to keep track of the availability and current prices of these cards, so that he/she can snap up any bargains.

To track these cards, the user executes a script that does the following.

1 5 0 retrieves the ongoing auctions from eBay on these cards;
0 retrieves the ongoing auctions from Yahoo! on these cards;
0 retrieves the ongoing auctions from uBid on these cards;
0 takes the user to the current price list at baseballcards.com;
0 takes the user to the current price list at Joescards.com; and
0 displays the latest issue of Trading Cards Weekly, an online newsletter.

Making User-customized Scripts. The MyScript application makes it possible for users to create their own scripts. These scripts can be stored in the central database and made available to all users. Alternatively a user can create an individual script database. The individual database capability reflects a peer-to-peer software structure, in which users can exchange scripts without going through the central database. This capability is useful for protecting scripts when privacy is an issue.

To illustrate script creation, suppose a user has just returned from a vacation in Paris. During the vacation, the user took photographs and recorded video footage. The user would like to share the vacation with ffiends. Unfortunately, all of them have pressing obligations that will prevent them from attending the user's slide show. With the MyScript

application, the user creates a script that will take friends through the highlights of the vacation.

To create a virtual scrapbook, a user creates a script of the following actions.

0 display a photo of the user atop the Eiffel Tower;
0 show the user's panoramic video of Paris, viewed from the Eiffel Tower;
0 0 take ffiends to the great online tour of the Louvre, as the user's film was
confiscated by the friendly gendarmes;
0 display more photos of the user's boat trip on the Seine;
0 take friends to the web sites of several famous boutiques on the Champs Elysee; and
0 conclude with a comparison of the current weather in Paris and in the user's hometown of Minot, South Dakota.

Finding the Right Script. Several advantages arise from the use of database technology to store and retrieve scripts, the most important advantage being the ability to find the right script for the task at hand. The MyScript application allows users to search for scripts by name, keyword, and other criteria in the same way as described above in connection with the team decision application. The MyScnipt application searches a central database of scripts (as well as the user's own personal script database).

Consider the task of upgrading a computer to run Windows 2000. The most frustrating and time-consuming task associated with that process is finding and downloading the correct device drivers for the computer (e.g., the driver for the display card, the network card, etc.). Manufacturers and computer information sites have web pages that give sets of instructions on where to find the drivers. Even better, however, is a script that visits the web sites and downloads the drivers automatically. For security reasons, it may be desirable that the MyScript application does not automatically install or execute new programs on a user's computer, though as an optional feature it may be automatically installed.

Similarly, suppose a user has a Sony Valo XYZ laptop and is looking for the new drivers. The user enters the MyScript application search screen and types in the
keyword search box "Sony Vaio XYZ laptop Windows 200O." The MyScnipt application generates a request that causes a search of the database and returns the top ten best matching scripts for the user's consideration. The user selects the script created by the "Sony Tech Support" user and tells the MyScnipt application to execute that script. Then the user can leave the room while the megabytes of driver files are downloaded to the computer.

Getting Advice on Scripting. Like the team decision application, the MyScript application is equipped with an Action Advisor functionality that can make suggestions about the availability of an appropriate script.

Suppose a user owns stock in several different companies. Every day, the user starts his/her computer, checks his/her positions in those stocks, and looks for company news associated with those stocks. At a user's request, the MyScript application stores records of all of his/her actions. After the user has checked the same set of stocks several times, the MyScript application generates an on-screen message explaining that

one or more scripts exist in the central database that may help the user simplify his/her task. The user can retrieve and execute an appropriate script to automate the daily stock routine.

Customized Script Applications. The possibilities for creating and usiner scripts are endless. These scripts can be customized to automate dynamic content creation in many different situations. Several examples are provided below.

*MyLife. Scripts that enable users to automate the acquisition and presentation of specified information (e.g., news, weather, stock prices, auctions, email) in a format appropriate for a particular type of equipment (e.g., PC or PDA).

+ MyAuctions. Scripts that enable users to search across sites for new auctions, to track expiring auctions, and to compare current bids on similar products in different auctions.

*MyHouse. Realtors record scripts to guide prospective buyers or renters through virtual home and community tours. Scripts can be tailored to appeal to different segments based on demographic and psychographic variables.

*MyLinks. As an alternative to simply listing links to other sites on a web page, users can create scripts that preview the set of links, and then automate progress through the links for another user.

*MyVacation. Users can create virtual scrapbooks that combine personally created content with publicly available content in multimedia formats.
A Business Process-oriented Application
The sophistication of Intemet-based applications has lagged behind the rapid increase in e-commerce activity. Software to streamline and automate business processes, and to enhance the productivity of workers faced with Internet-sized resources, has tended to emphasize single-function and within-organization processes.

This focus falls to capitalize on the advantages of a market structure in which needs may be outsourced in a widespread and highly competitive environment. The system and method of the present invention enables seamless, integrated execution of standardized business processes, using software templates that can be communicated both within and between organizations.

An important goal of Internet use for many businesses is to use the power of the Internet to reduce operating costs. A good example of this is trying to streamline the purchasing process. Two problems exist with currently available technology solutions for business processes. First, the "surfing" approach to Internet use, as illustrated by browser capabilities, is not well suited to these goal-directed, complex sequences of operations. Second, in many business that are beginning to integrate the Internet into their repertoire of business practices, the people who have to carry out the tasks in the Internet environment often do not have the technical skills to take advantage of the Internet's capabilities. These issues mean that the business processes must be simplified.

A script-based approach can be very useful for many business practices.

Appropriate applications that embody complex and repetitive business practices can be created and used according to the present invention. The

actions in the application WO 01/16765 PCTIUSOO/23749
25
may return results of search matches in the thousands, but there is no oruarantee that the results will provide a set of sources with informational utility (i.e., high quality, relevant inforination) for the manager's task goals.

With the system and method of the present invention, the purchasing manager can retrieve a script from the database to automate the sequence of activities. Once initiated, the script automatically searches a list of approved sites for a product. Then the script generates and sends a purchase order to the selected supplier. An internal record of the purchase is also created when the script sends a message to the appropriate budget department to log the completed purchase. The script streamlines the purchase process in two ways: 1) by automating repetitive aspects of the task, and 2) by keeping the amount of inforination to be evaluated (e.g., potential suppliers) at a manageable level. These actions can all be performed by existing software, but the system and method of the present invention automate the process.

A Portable Device-oriented Applicatio
Next-generation wireless capabilities will enable portable digital assistants (PDA's), cellular phones, and other wireless devices to connect with the Internet at all times. The small screen and the pen-based user interface typically used on these devices make doing non-trivial work on these devices quite difficult. In addition, there is a significant challenge involved in making user interfaces in these devices look even remotely similar to what appears on a standard desktop or laptop computer (efforts such as UIML partially address this user interface issue).

The system and method of the present invention allow a different, more powerful approach. Instead of trying to force a desktop interface onto a PDA, the present invention focuses on the task that the user is trying to accomplish. The same scripts are used on PDA's as on desktops; the user is not forced to scroll through large pages to click the right button or try to type on the PDA. Instead, the user can Just select the right script for execution, and the front-end application takes over script execution.

An even more powerful feature is that the front-end application can interpret the script slightly differently on the PDA than on the desktop. The actions executed are the same on each platfonn, but the displays can be tailored to each device. In this way, the user gets the same actions on every platform running the front-end, but the display is always appropriate for the platform.

FIG. 14 shows as an example a script that a user activates to monitor a stock and contact a broker. FIGs. 15 and 16 illustrate how the script is processed differently I 0 by different front-end applications. The script shown in FIG. 14 has four actions.

Action I gets a stock price for the XYZ Corporation from a web site, Action 2 gets news on the XYZ Corporation, Action 3 establishes a communication link with the user's broker, and Action 4 sends a short message confirming the communication to the broker.
With reference to FIG. 15, the script (called Script I in FIGs. 15 and 16) is processed in one manner by a Device I that executes Application 1. The operations performed by Application I when executing Actions 1 -Actions 4 are represented as 0104, respectively. For example, Device I

is a desktop computer with a fast Internet connection. In this case, Application I executes Action I to bring up a web page with a stock quote and graphical charts displaying details regarding the stock price of the XYZ Corporation. Action 2 sequences through a set of web pages containing the latest news reports on the XYZ Corporation. Action 3 initiates a video conference with the user's broker. Action 4 prompts the user for keyboard input for a message to be sent to the broker.

By contrast, as shown in FIG. 16, when Script 1 is executed by Application 2 on Device 2, different operations are performed, referred to as Operations 01', 02', 03' and 04', respectively. For example, Device 2 is a PDA with perhaps a slower wireless Internet connection, a smaller display screen, and a pen-based input (no keyboard.), the actions must be interpreted differently to be effective. Application 2 executes Action 1, which brings up a web page with a stock quote, but does not download the graphical charts automatically. This avoids the significant delay involved in downloading graphics over a slow connection, but leaves the user the option of downloading the charts if desired. Action 2 displays a set of headlines, rather than sequencing through a set of web pages; in this format, the user only downloads a story if it is of interest, again avoiding delays associated with unnecessary downloads. Action 3 initiates an audio conference with the user's broker, because a slow link may not support a twoideo connection, and the screen may not provide the resolution required to make
way vi I
such a link useful. Action 4 prompts the user for pen-based (rather than keyboard) input for a message to be sent to the broker. Thus, device characteristics, such as user interface features and communication bandwidth, are considered when specifying how a front-end application that executes on that device will interpret a script action.

Content may be adapted (i.e., translated or modified) into a format compatible with device characteristics and/or specific actions may be modified, deleted or augmented according to device characteristics.

The front-end application on each platform interprets the actions of a script in a manner suitable for that platform. This can be done because the actions indicate what is to be done, rather than how to perforin the action. For example, the script specifies that a communication link is to be established with the broker, but it does not specify how to establish that link or what type of link to establish. The front-end application detennines how to establish the link and what type of link to use, based on the type of device on which it is designed to execute.

A Sight Impaired-oriented Application
Currently, sight-impaired users of the WWW must rely on tools such as textonly browsers. Unfortunately, most WWW sites have many unlabeled graphical images, video streams, and other features that cannot be translated automatically into text. This creates a severe bam'er-to-use for the sight-impaired, and for the companies that employ them. Modifying, testing, and maintaining existing WWW sites to allow for use by these text-based browsers can be extremely costly. This cost is not only in the direct dollars spent, but also in delays to market for new WWW sites, as well as reductions in the effectiveness of the WWW sites for non-impaired users.

The system and method of the present invention provide an alternative that overcomes these drawbacks by empowering sight-impaired (and other disabled users) through script functionality. By providing users with appropriate scripts, the system and method of the present invention,

combined with appropriate front-end applications, I 0 can completely avoid the need to rework VVW-W sites. These sites are provided with front-end applications, but once a W@ site has taken that step, all that is required is to provide the appropriate scripts for execution. These scripts translate the online content into formats that are compatible with the user's constraints. Because scripts can be purely text-based, the sight-impaired user can function without disadvantage in 1 5 this envirom-nent. In fact, by providing streamlined, task-oniented access to a site, such users may achieve a competitive advantage.

As an example, consider the XYZ Corporation's WWW site on which clothing is sold. To be attractive to its target population, this WVV'W site has many flashy graphics, pictures, and videos to attract and hold viewer interest. This WWW site is a largely graphical WWW site with very little text. Navigating and usiner this WWW site
In
is extremely difficult for the visually impaired. Changing this WWW site to be accessible to visually impaired users may be expensive in both money and time, with the result still of minimal functionality to the visually impaired.

Like most consumer WWW sites, the XYZ Corporation's WWW site is based on a database system that maintains a list of the inventory with textual descriptions of items, prices, and other information. In addition, again like most WWW sites, ultimately the user provides information with product numbers in a shopping cart fonnat and then enters in billing and shipping information. By providing scripts and a front-end application designed to interact with visually impaired users on a device, the site can be made efficiently useful for the visually impaired.

For example, any user may wish to (1) search for desired apparel, (2) get information on that apparel, and (3) purchase the apparel. Any of these tasks can be accomplished by sending requests to the database of product information at the XYZ Corporation VIWW site and then interpreting the results - actions that take place on the graphical WWW site, but that are obscured by graphically-oriented HTML pages.

When XYZ Corporation provides a script to "search for apparel," a script to "retrieve information on apparel," and a script to "purchase apparel," a visually-impaired user can shop on this WWW site effectively. The front-end application on the user's device is designed to interpret the scripts in a fashion useful for the visually impaired. The front-end applications that interpret the scripts may be downloaded by users from a XYZ's site or at central sites that store scripts for sight-impaired users. A variety of front-end applications may be available, each designed for use on different types of devices (desktops, PDA's, etc.) by sight-impaired users. For example, data returned to the user are processed by the front-end application using computer-generated (synthesized) speech rather than supplying text and images to the screen. Upon entry to a WWW site, the visually impaired user would be presented with a list of scripts (or alternatively, could search for scripts in the script database) if the user does not already have them. Examples of these scripts are explained below.

The "search for apparel" script includes an action that the front-end application executes to computer-generate (synthesize) speech that is coupled to a loudspeaker on the user's device in order to prompt the user for keywords on which to search. User input or responses can be by keyboard input or by voice response. For the latter case, a microphone associated with the user's device detects the user's spoken keywords, and

a script action causes the front-end application to translate the spoken keywords into a search query that is sent to XYZ Corporation's database server. After execution of the search, the results are returned, and the front-end application executes a script action that inforrns the user (in computer-orenerated speech) of the names of any garments that were found. If requested by the user (via keyboard or voice), this front-end application executes a script action to activate the "retrieve inforination on apparel" script for a selected product (e.g., a particular garment).

The "retrieve information on apparel" script includes an action that is executed by the front-end application to generate and send a request to the XYZ database server to retrieve information about the selected product, such as the appearance, price, and 'lability of the garment. The fTont-end application executes a script action that
aval
reports this inforrnation to the user in computer-generated speech. The front-end application then executes a script action to -by voice prompt- give the user the option to add the item to a shopping cart or activate the "purchase apparel" script.

The "purchase apparel" script allows the user to purchase all garments in his/her shopping cart. This script includes actions to solicit from the user information on billing and shipping. It does so by prompting the user for each piece of information required using computer-generated speech. The user enters the information either by voice or by keyboard. Note that if the user has been to the site before, this script can be considerably simplified if the inforination has been stored by the WWW site.
Note that each site will likely have to contain several scripts to enable effective use by visually impaired visitors (though given the commonality of software currently in use, such scripts may be transferable and usable across different sites). This is, however, a much less expensive task than retrofitting and updating existing, expensive WWW sites. Moreover, the scripts can be written such that each script can perform any one of a class of actions to enable a sight-impaired user to interact with a multitude of different sites. For example, one script is directed to converting a web page from text and images to text and synthesized voice that can be used with many different web sites for many different applications, another script is directed to receiving voice responses from a user and converting them to text strings that are sent to a web site, etc.

The portable-device oriented application and the sight-impaired application are examples of the flexible nature of the system and method of the present invention. In general, a front-end application can be written to perforin a variety of operations that are designed to be compatible with device characteristics and/or intended user characteristics. For example, a front-end application may interpret a script to present content (text, video, audio, or a combination thereof) to a user or receive user input (keyboard, pen-input, voice input, mouse device input) in a fonnat compatible with characteristics of the device. Similarly, a front-end application may interpret a script to cause the device to perform operations based on modified script actions that are best suited to that device (or that class of devices), or to an intended user of that device.

Moreover, a front-end application of a device may interpret a script to present content or to receive input from a user in a mode compatible with characteristics of an intended user (sight-impaired user, hearing

impaired user, physically handicapped user, etc.) of the device.

5 To summarize, the present invention is directed to a system and method for sharing scripts of device actions. Scripts representing at least one device action are stored in a scripts database. Devices access the scripts database to perform operations based on one or more scripts retrieved from the scripts database. The script actions may be actions involving access to or use of the Intemet/WorldWideWeb. In each device, a (front-end) application is provided that interprets the script to perfon-n device operations.

The above description is intended by way of example only and is not intended to the limit the present invention in any way.

Claim

1 A method for sharing scripts of device actions comprising steps ofstoring scripts representing at least one action that can taken by a device in a
scripts database; and
accessing the scripts in the scripts database from a device to perform operations on the device based on one or more scripts retrieved from the scripts database.

2 The method of claim 1, wherein the step of storing the scripts comprises transmittincr the scripts to a database that is remotely located from the device on which
the scripts were generated.

3 The method of claim 1, wherein the step of accessing the scripts comprises generating a search query that comprises one or more search terms for
submission to the database.

4 The method of claim 3, wherein the step of generating a search query comprises using one or more of a plurality of search criteria, including date, user name,
and task to perform.

5 The method of claim 1, and further comprising the step of generating scripts on a computer device for storage in the database.

6 The method of claim 1, wherein the step of generating scripts comprises recording computer actions that involve access and use of information from the
Internet.

7 The method of claim 1, and further comprising the step, on a first device, of interpreting a script defining at least one action to perforin an operation, and on a second device, interpreting the script defining the at least one action to perform an operation that is different from the operation perfon-ned by the first device.

8 The method of claim 1, and further comprising the steps ofmonitoring actions of a user of a device with respect to the use of scripts; and suggesting a script to a user based on characteristics of scripts accessed or used by a user.

9 The method of claim 8, wherein the step of monitoring actions comprises steps of storing a list that identifies scripts recently executed by a user; searching the scripts database to locate scripts that have at least one characteristic which matches that of the scripts in the

list; and wherein the step of suggesting
comprises returning at least one matching script to the user.

10 The method of claim 9, wherein the step of suggesting comprises returning a list of matching scripts ranked according to criteria.

11 The method of claim 9, and wherein the step of storing a list comprises storing in a script access database information representing which scripts a user frequently accesses, the frequency of script usage, the number of times a user rejected a
script for execution when suggested to the user.

12 The method of claim 8, wherein the step of monitoring comprises recognizing that a user performs the same set of actions at a predetermined time of day and/or frequency; storing a script representing those actions in the scripts database.

13 The method of claim 12, wherein the step of suggesting comprises suggesting a script representing actions that were determined to have been performed
by the user on a consistent basis.

14 The method of claim 13, and further comprising the step of setting an automatic timer associated with the script so that the script is automatically activated at
a predetermined time and/or frequency.

15 The method of claim 1, and further comprising the step of transmitting a message containing a script or a script identifier from a first user of a first user device
to a second user of a second user device.

16 The method of claim 15, wherein the step of transmitting further comprises including a unique user identifier of the first user with the script, whereby the second user of the second device must enter the unique user identifier to validate the first user before executing the script.

17 The method of claim 15, and further comprising storing a list of unique 'dent'fiers for each user of a device.
user 1 1

18 The method of claim 15, and further comprising encrypting the message prior to transmission.

19 The method of claim 1, wherein one or more of the scripts represents actions involving access to or use of the Intemet/WorldWideWeb.

20 The method of claim 19, wherein the step of storing scripts comprises storing scripts that represent actions selected from the group consisting of. opening a web page from a web site; opening web pages each from a plurality of web sites; searching a web site for information such as music, video or other content; downloading information from a web site such as music, software, video or other content; comparing select information from multiple web sites; and creating person profile infortnation for a user such as user registration and on-line shopping information, playing a list of music tracks from one or more web sites.

21 The method of claim 19, wherein the step of storing scripts comprises storing a script representing actions to: retrieve auction information of

a particular product, service, class of products or services from each of a plurality of different web sites; and retrieve pricing information on the products, services or class of products or services from a each of a plurality of different price-listing web sites for those products, services or class of products or services.

22 The method of claim 19, wherein the step of storing scripts comprises storing a script representing actions to: search one or more web sites of suppliers for pricing information on products or services; generate and transmit a purchase order to one of the suppliers based on the pricing inforniation obtained; maintain an internal record of the purchase.

23 The method of claim 22, wherein the step of storing scripts comprises further storing a script to transmit a message containing the internal record of the purchase to an appropriate budget department of a business entity.

24 The method of claim 1, and further comprising the step of, in a device, interpreting a script to present content to a user or receive input from a user in a format ible with characteristics of the device.
compati I 1

25 The method of claim 1, and further comprising the step of, in a device, interpreting a script to cause the device to perform operations based on modified script actions.

26 The method of claim 1, and further comprising the step of, in a device, interpreting a script to present content or receive input from a user in a mode ible with characteristics of an intended user of the device.
compati 1 1

27 A system for sharing scripts representing computer actions, comprising: at least one database server computer that stores a database of scripts, each representing at least one action that can be taken by a device; and a plurality of devices that communicate with the database server computer to access one or more scripts in the database to perform device operations based on one or more scripts.

28 The system of claim 27, wherein each device comprises a processor that executes at least one front-end application.

29 The system of claim 28, and further comprising at least one middleware computer that executes a executes a middleware program that interprets requests and actions received from the front-end application of a device, formulates queries to the database server, and receives and forwards scripts from the database server for action by a front-end application on a device.

30 The system of claim 28, wherein the database server computer executes a middleware program that interprets requests and actions received from the front-end application of a device, formulates queries to the database, and receives and forwards scripts from the database for action by a fTont-end application on a device.

31 The system of claim 28, wherein a first front-end application on a first device interprets a script defining at least one action to perform

an operation, and a second front-end application on a second device interprets the script definina at least one action to perform an operation that is different from the operation performed by the first device.

32 The system of claim 3 1, wherein the first device is a portable or hand
held device and the second device is a desktop or laptop device.
33 The system of claim 27, wherein at least one of the devices generates scripts that are transmitted for storage by the database server.

34 The system of claim 27, wherein at least one device generates scripts by recording computer actions that involve access and use of infori-nation from the
Internet.

35 The system of claim 29, wherein the front-end application of a device or the middleware program monitors actions of a user with respect to the use of scripts and suggests a script to a user based on characteristics of scripts accessed or used by a
user.

36 The system of claim 35, wherein the front-end application or the middleware program monitors actions by: storing a list that identifies scripts recently executed by a user; searching the scripts database to locate scripts that have at least one characteristic which matches that of the scripts in the list; and wherein the front-end application or middleware program returns at least one matching script to the user.

37 The system of claim 28, wherein the front-end application of a device interprets a script to present content to a user or to receive input from a user in a format
compatible with characteristics of the device.

38 The system of claim 28, wherein the front-end application of a device interprets a script to cause the device to perform operations based on modified script
actions.

39 The system of claim 28, wherein the front-end application of a device interprets a script to present content or receive input from a user in a mode compatible with characteristics of an intended user of the device.

40 The system of claim 28, and further comprising a plurality of database server computers, each storing a database of scripts, and a plurality of middleware computers that communicate with the devices and with the database server computers, each middleware computer interpreting requests received from the front-end application of a device, fonnulating queries to a database server computer, receiving a script from a database server computer, and forwarding the script for action by the front-end application on a device.
41 A method for automating tasks involving access to or use of information from the Intemet/WorldWideWeb, comprising steps ofstoring scripts each representing at least one action involving access to or use of
information from the Internet/WorldWideWeb in a database;
accessing scripts in the database from at least one device to perfort-ii actions on
the device; and
in the device, interpreting the scripts to perform device operations

based on at
least one action defined by a script.

42 The method of claim 41, wherein the step of interpreting comprises interpreting a script to present content to a user or receive input from a user in a format
compatible with characteristics of the device.

43 The method of claim 41, wherein the step of interpreting comprises interpreting a script to cause the device to perform operations based on modified script actions.

44 The method of claim 41, wherein the step of interpreting comprises interpreting a script to present content or to receive input from a user in a mode ible with characteristics of an intended user of the device. compati

45 The method of claim 41, and further comprising the step of providing a front-end application in each of a plurality of devices, wherein the front-end application in each device interprets scripts to perfortu operations on the corresponding device consistent with characteristics of the corresponding device.

46 The method of claim 41, and further comprising the step of providing a front-end application in each of a plurality of devices, wherein the front-end application in a device interprets scripts to perform operations on the corresponding device
consistent with characteristics of a user of the corresponding device.

47 A system for automating tasks involving access to or use of information
from the Intemet/WorldWideWeb, comprising:
at least one database server computer that stores a database of scripts each representing at least one actions involving access to or use of information from the
Intemet/WorldWideWeb; and
a plurality of devices that communicate with the database server computer to access one or more scripts in the database, wherein each device comprises a front-end application that interprets the scripts to execute device operations based on script
actions.

48 The system of claim 47, wherein the front-end application of a device interprets a script to present content to a user or to receive input from a user in a format
compatible with characteristics of the device.

49 The system of claim 47, wherein the front-end application of a device interprets a script to cause the device to perform operations based on modified script
actions.

50 The system of claim 47, wherein the front-end application of a device interprets a script to present content or to receive input from a user in a mode compatible with characteristics of an intended user of the device.
?